Introduction to Software Design with Java

Martin P. Robillard

Introduction to Software Design with Java

Springer

Martin P. Robillard
School of Computer Science
McGill University
Montreal, Québec, Canada

© 2017 Circlecreativestudio/iStock

ISBN 978-3-030-24093-6 ISBN 978-3-030-24094-3 (eBook)
https://doi.org/10.1007/978-3-030-24094-3

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This book is inspired by well over a decade of teaching software design at McGill University. At first, my focus was to explain the software design know-how available from high-quality references. Soon, however, I realized that the main challenge of teaching software design lay elsewhere. Communicating *how* to apply a design technique or use a programming language mechanism was relatively easy. The real struggle was to convey in which *context* we want to use a certain design technique, and *why*. To do this, I needed to explain what is going on in a software developer's head. Over time, my lectures came to be more about exploring the space of alternative design decisions one can make in a given context.

The goal of this book is to help readers learn software design by discovering the *experience* of the design process. I share my experience of designing software through a narrative that introduces each element of design know-how in context, and explores alternative solutions in that context. The narrative is supported by hundreds of code fragments and design diagrams.

My hope is that this book can serve as an effective resource and guide for learning software design. However, I do not believe that it is possible to develop significant design skills solely by reading a book. In my own learning process, I have benefited hugely from reading other people's code, regularly writing code, and relentlessly refactoring existing code to experiment with alternative design solutions. For this reason, this book places a lot of emphasis on coding and experimentation as a necessary complement to reading the text. To support this aspect of the learning process, I provide a companion website with practice problems, and three example applications that capture numerous design decisions. An orientation through these example applications is provided in a section called "Code Exploration", at the end of each chapter.

As its title indicates, this book provides an introduction to software design using the Java language. The code used throughout the book, as well as the example applications, are in Java (version 8). My use of the Java language, however, is a means to communicate design ideas, and not the topic of the book. I aimed to cover design concepts and techniques that are applicable in a host of technologies. Many concepts (such as encapsulation), will be generally relevant in any technology. Others (such as inheritance) will be paradigm-specific, but usable in multiple programming languages. For both general and paradigm-specific information, it should be possible to adapt the examples to other languages relatively easily. In a few cases, the material needs to address a Java-specific mechanism with implications on design (e.g., cloning). In such cases, I make sure to present the mechanism as one implementation of a more general technique.

This book is targeted at readers who have a minimum of programming experience and want to move from writing small programs and scripts to tackling the development of larger systems. This audience naturally includes students in university-level computer science and software engineering programs. However, I kept the prerequisites to specific computing concepts to a minimum, so the content is also accessible to programmers without a primary training in computing. In a similar vein, understanding the code fragments requires only a bare minimum knowledge of the language, such as would be taught in an introductory programming course. Information about Java that is crucial to understand the text is provided in an appendix, more advanced features are introduced and explained as necessary, and I make a minimum of references to specific elements of the language's class library. My hope is thus that the book can be useful to almost anyone who wants to write clean, well-designed software.

Organization of the Book

The first chapter is a general introduction to software design. The subsequent chapters provide a progressive coverage of design concepts and techniques presented as a continuous narrative anchored in specific design problems. In addition to the main content, the book includes different features to orient readers and help use the book as a launchpad for further exploration and learning.

- **Chapter Overview:** At the beginning of each chapter, a callout lists the concepts and principles, programming mechanisms, design techniques, and patterns and antipatterns covered in the chapter.
- **Design Context:** Following the overview, a paragraph titled "Design Context" introduces the design problems that are used as running examples in the chapter. It is thus not necessary to read all previous chapters to understand the code discussed in a given chapter.
- **Diagrams:** Each chapter includes numerous diagrams that illustrate design ideas. Although they are provided to illustrate the ideas in the text, the diagrams are also realistic illustrations of diagrams that can be used in practice as part of design discussions.
- **Code Fragments:** Each chapter includes many code fragments. The code generally follows the conventions presented in Appendix B, with occasional concessions made to make the code more compact. A complete version of the code fragments can be downloaded from the companion website (see below).
- **Insights:** In each chapter, the main numbered sections are followed by an unnumbered section titled "Insights". This section forms an actionable summary of

the key information and advice provided in the chapter. It is meant as a sort of catalog of applicable design knowledge, and assumes the material in the chapter has been mostly assimilated. The insights are in bullet points to be easily perused.

- **Code Exploration:** Following the "Insights" section is a section titled "Code Exploration" that provides a discussion of software design in practice. To facilitate good flow and avoid getting lost in details, the design contexts discussed in the main chapters are kept as simple as possible. As a result, some interesting aspects of the software design experience do get lost in the simplification. The code exploration activity supported by this section is the opportunity to consider how some of the topics presented in the chapter manifest themselves in reality. The "Code Exploration" section points to specific parts of the code of the example applications. Before reading the text in the Code Exploration section, I recommend reviewing the code referenced and trying to understand it as much as possible. The example applications discussed in the "Code Exploration" section are described in Appendix C. They include JetUML, the application used to create all the diagrams in the book.
- **Further Reading:** The Further Reading section provides pointers to references that complement the material presented in the chapter.
- **Companion Website** Additional resources for this book are available in the repository `https://github.com/prmr/DesignBook`. The material in the repository includes a complete and commented version of the code that appears in the chapter content, as well as practice exercises and their solutions.
- **Example Applications** The three Java applications described in Appendix C were developed following many of the principles and techniques described in the book, and are provided as an accessible basis for additional study and exploration.

Acknowledgments

I am most grateful to Mathieu Nassif, who carried out a detailed technical review of the entire manuscript, providing me with hundreds of corrections, suggestions, and interesting points for discussion. I warmly thank Jin Guo for reviewing most of chapters and testing some of the material in her own teaching, and Kaylee Kutschera, Brigitte Pientka, and Clark Verbrugge for feedback on various parts of the manuscript. I am also thankful to Ralf Gerstner, the executive editor in charge of computer science at Springer, for believing in the project from the start and for seeing it through with his usual diligence and professionalism.

Martin P. Robillard
April 2019

Contents